第 33 卷 第 6 期 计 算 机 工 程 2007 年 3 月
Vol.33 No.6 Computer Engineering March 2007
·软件技术与数据库· 文章编号：1000—3428(2007)06—0069—03 文献标识码：A 中图分类号：TP311

基于代理的 3PC 分布式事务提交协议

鄢喜爱[1,2]，杨金民[1]，张波云[2]，常卫东[2]
(1. 湖南大学软件学院，长沙 410082；2. 湖南公安高等专科学校计算机系，长沙 410138)

摘 要：原有 3PC 分布式事务提交协议能克服协同者发生故障而有可能产生事务阻塞问题，但其开销大。该文提出了一种基于代理的 3PC 事务提交协议，该协议通过增加协同者的代理节点，使得参与者相信协同者是正常的，从而不必关心新协同者的选举问题，能降低 3PC 无故障时的额外开销。

关键词：代理；三阶段提交协议；分布式事务

Distributed Transaction 3PC Protocol Based on Proxy

YAN Xiai[1,2], YANG Jinmin[1], ZHANG Boyun[2], CHANG Weidong[2]
(1. Software College, Hunan University, Changsha 410082;
2. Department of Computer Science, Hunan Public Security College, Changsha 410138)

【Abstract】3PC commitment protocol can overcome the blocking problem caused by coordinator fault, but it brings extra overhead. This paper proposes a 3PC protocol based on proxy, in which the participants believe that the coordinator is fault-free by constructing proxy of coordinator. In the protocol, the participant need not consider to vote new coordinator, lowering the overhead of the original 3PC.

【Key words】Proxy; 3PC; Distributed transaction

分布式数据库系统是物理上分散而逻辑上集中的数据库系统。由于其具有分布式的特点，事务的原子性、一致性、隔离性、持久性(ACID)4 个属性更加难以得到保证。分布式事务要求分布在各个节点执行的子事务要么全部提交，要么一个都不提交，保证数据库的状态总是从一个一致的状态变迁到另一个一致状态。为此，事务必须执行一个提交协议。

Gray在 1978 年提出了两阶段提交协议(2PC)[1]，它通过两个阶段完成事务提交，但有可能发生事务阻塞。为了解决事务阻塞问题，Skeen在 1981 年提出了 3 阶段提交协议(3PC)[1]，在 2PC中插入了一个预提交阶段，但增加了很大的额外开销。由于在分布式系统中，代理和冗余是常用来增强可用性的技术，本文提出了一种在代理环境下，既能防止事务阻塞，又能节省额外开销的协议——基于代理的 3PC事务提交协议(Proxy Based 3PC，P3PC)。

1 基于代理的 3PC 分布式事务提交协议

1.1 原有 3PC 分布式事务提交协议分析

第 1 阶段：协同者将记录<begin_commit>加到日志中，向所有参与者发<Prepare>消息，启动 Timer 后进入等待；参与者收到<Prepare>消息后，如愿意提交属于它的那部分，则向协同者发<Ready>消息，如因某种原因不愿提交，则向协同者发<Abort>消息，记入日志。

第 2 阶段：如果所有参与者均回答<Ready>，协同者向所有的参与者发<Global_Commit>消息，否则，向参与者发Global_Abort>命令；如 Timeout 超时，也向参与者发<Global_Abort>命令，并记入日志。参与者根据协同者的命令提交事务，或撤消事务，向协同者发送<Acknowledge>消息,记入日志。协同者等待所有参与者的<Acknowledge>消息，记入日志后结束事务。两阶段提交过程如图 1 所示。

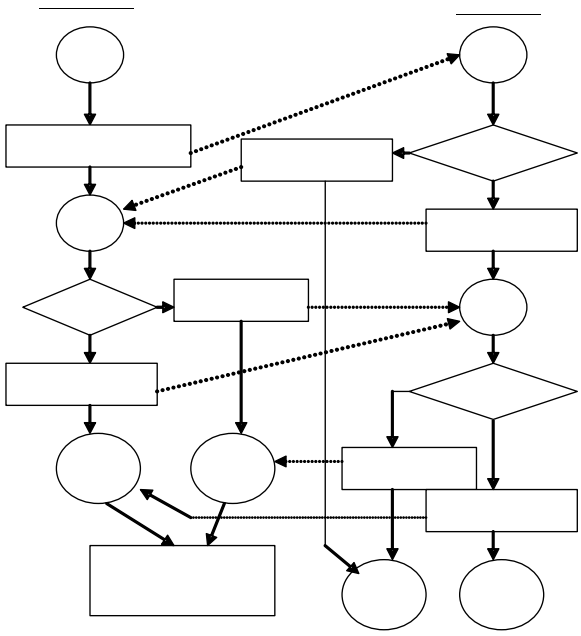

图 1 两阶段提交协议的活动

2PC 是一种简单、低开销的提交协议，但有一个弱点：当参与者处于<ready T>状态时，协同者发生故障，并且至少有一个参与者出现故障，这时终结就不可能，因为可工作的

基金项目：国家自然科学基金资助项目(NSFC 60473031)；湖南省自然科学基金资助项目(05JJ30116，04JJ6032)

作者简介：鄢喜爱(1972－)，男，硕士生，主研方向：分布式数据库；杨金民，博士、副教授；张波云，博士生、副教授；常卫东，硕士生、讲师

收稿日期：2007-03-27 **E-mail：**yanxiai222@yahoo.com.cn

参与者无法知道出故障的参与者是否作出提交的决定，如果通过选举产生新的协同者来指挥行动，可能产生不一致的状态。这时参与者必须等待协同者的恢复，如果短时间内不能恢复，就会导致事务阻塞。如果使用了锁，还会导致其它事务也被迫待等待。

事务提交非阻塞的充要条件[2]：(1)没有状态同时与提交状态和撤消状态相邻。(2)没有不可提交状态与提交状态相邻。

原 3PC 协议通过在<wait>(以及<ready>)和<commit>状态之间增加一个额外的预提交状态来实现，它作为一个缓冲，用于在准备提交但还没有提交的时候，以满足事务提交的非阻塞条件，其状态如图 2 所示。

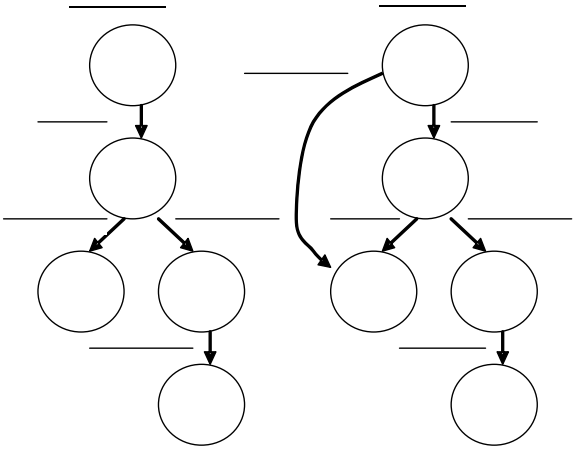

图 2 3PC 中协同者和参与者的状态

当出现 2PC 中那种阻塞情况时，由于出故障的参与者还未作出提交的决定，参与者可以执行终止协议且唤起选举协议选举一个新的协调者，可以保证事务的原子性。但是，在 3PC 正常情况下，参与者不仅需要实时记录协同者通信以及本地事务的执行，还要实时存储并更新其它参与者的信息列表，以便在协同者超时的情况下，发起新的协同者选举。这样开销代价大，加之实践中阻塞发生的可能性不大，这种额外开销很不合算。

1.2 基于代理的 3PC 协议的工作机制

1.2.1 P3PC 协议的模型

为了防止事务阻塞，P3PC 协议以传统的 3PC 提交协议为基础构建；为了降低开销，P3PC 协议通过增加一个协同者代理节点，使得参与者相信协同者是正常的，将不采用选举协议选举新的协同者，其模型如图 3 所示。

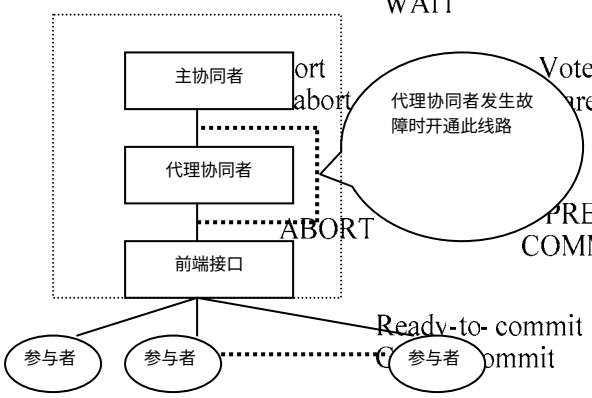

图 3 P3PC 协议的模型

1.2.2 P3PC 正常情况下的运行机制

P3PC 正常工作情况下，参与者并不直接与主协同者进行消息交换，而是与代理协同者进行消息交换，代理协同者类似于WWW代理服务器，本身具有自治性，它扮演事务协同者的角色，并实时(小于参与者设定的 timeout 时间)将产生的日志传播给主协同者。主协同者拥有代理协同者同样的功能，但不指挥行动，主要负责将接收的日志来重建一个与代理协同者版本相同的数据库。代理协同者与主协同者之间数据的同步转发对于参与者来说是透明的，即对于参与者来说，它们和原有的 3PC 一样，只知道有一个协同者存在。

1.2.3 P3PC 在协同者有故障时的运行机制

在网络通信中，代理协同者与主协同者之间每完成一次日志消息的同步转发之前要建立一次连接，假设不存在网络分割和报文丢失，如果连接不成功，可判定对方出现故障[3]。

当代理协同者发生故障时，可以通过接口机制直接响应主协同者，由主协同者接管，使参与者感觉不到服务发生了中断。当代理协同者恢复时，主协同者为代理协同者提供一个正确的数据库，该数据库能反映事务在代理协同者无法工作时所作的更新，这将使代理协同者恢复后能够重新开始正常的工作。

当主协同者发生故障时，由于代理协同者本身具有自治性，它同样可完成剩下的工作。当主协同者恢复后，代理协同者同样向主协同者提供一个正确的数据库，然后继续工作。

此协议只能容代理协同者或主协同者之间的一个故障，也可能存在最悲观的情况，即主协同者与代理协同者都发生故障。但是，这种情况发生的可能性非常小，其理由是：

(1)故障检测的时间间隔非常短(它是协同者之间每次传送数据建立连接的时间间隔)。

(2)由于系统中存在一个协同者的副本,当出现一个故障时，通过复制，出故障的协同者恢复时间是短暂的[4]。

此机制可以在绝大多数情况下满足系统中的可用性要求。为了建立更高的可用性系统，考虑系统的可扩展性，使用动态的多级代理，这是我们下一步的研究方向。

在这种机制下，参与者可以认为协同者是不会失败的。参与者不必再通过选举产生新的协同者，从而也就不要相互保存其它参与者的信息列表。

1.3 P3PC 协议的算法描述

P3PC 的算法如下[5]：

MAIN COORDINATOR:

Step1 实时与代理协同者建立一次连接，如果连接超时，接管代理协同者的行动，等待代理协同者恢复后，又执行 Step1。

Step2 接收代理协同者产生的日志消息

Step3 发送本次传送结束消息。

PROXY COORDINATOR:

Step1 投票

在日志中写入“ begin_commit ”，发送“ prepare ”指令给所有参与者，启动 Timer 后进入等待；如果超时，转 step2B。

step2A 预提交

如果收到所有参与者的“ Ready ”回应，写“ preare_Commit ”到日志，发送“ preare_Commit ”命令给所有参与者。

step2B 全局取消

如果收到至少一个参与者的“ Abort ”回应，或者参与者超时，写“ Global_Abort ”到日志中发送“ Global_Abort ”命令给参与者转 step4。

step3 全局提交

等待，直到收到所有参与者的“ 预提交确认 ”回应，写

“ Global_Commit ”到日志中，发送“ Global_Commit ”到参与者。

step4 终结

等待，直到收到所有参与者的确认。

PARTICIPANTS:

Step1 投票

如果参与者准备提交，则发送“ Ready ”回应给协同者，否则发送“ Abort ”回应，等待协同者的指令。

Step2A 预提交

如果收到“ preare_Commit ”指令，则转 Step3 等待全局提交。

Step2B 取消

执行本地事务放弃处理。

Step3 提交

等待，直到收到“ Global_Commit ”命令，执行本地提交处理。

Step4 终结

给协同者发送确认。

2 P3PC 与 3PC 的通信开销比较

从算法描述可知，P3PC 与 3PC 中决定提交和决定夭折的消息交换相同，所以只需分析为了在协同者发生故障能够恢复而产生的额外通信开销。

在分析比较中，假设协同者和参与者节点个数为 n≥3，网络拓扑为任意，位置信息列表的实时更新时间间隔与协同者之间传递数据时间间隔相同。

在原有的 3PC协议中，为了能在协同者发生故障时，能通过选举产生新的的协同者，任意一节点要关心其余所有节点的位置。节点要通过消息交换获取位置情况，称为位置跟踪消息。这种跟踪消息是维持正常工作的额外开销。一个节点要跟踪另外一个节点的位置，通过向另外一个节点发送位置查询消息，然后返回一个位置消息，即一次跟踪须交换两条消息[6]。

任意两个节点之间要相互获取位置，需交换 4 次跟踪消息，对于任意网络拓扑的 n 个节点，需交换 2n(n-1)条消息。图 4 是 3 个节点的位置跟踪情况，需要交换 12 条消息。

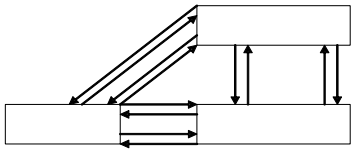

图 4 原 3PC 中 3 个节点的位置跟踪消息交换

在 P3PC 协议中，增加了一个代理节点，一共有 n+1 个节点完成事务。P3PC 的额外开销有两个，一个是代理协同者与主协同者的消息交换，另一个是位置跟踪消息交换。

代理协同者与主协同者完成一次数据传递如图 5 所示，一般需要完成 6 次消息交换[7]。

对于位置跟踪，应将协同者和参与者分开。因为参与者相信协同者是正常的，可以不关心其它参与者的位置。每个参与者只需与代理协同者之间进行位置跟踪消息交换，每个参与节点与代理节点相互交换的位置跟踪消息为 4 条，所以总位置交换消息是 4(n-1)条。图 6 是 3 个节点的位置跟踪消息交换情况，需要交换 8 条消息。

两种协议的额外开销中的消息交换总数的比较如表 1，容易看出，当 n 超过 4 时，P3PC 的额外开销就比 3PC 要小，而且随着 n 的取值越大，它们的差别越大，越节省开销。

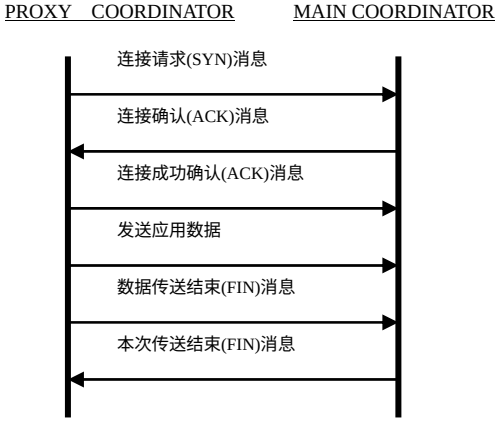

图 5 P3PC 中代理协同者与主协同者消息交换

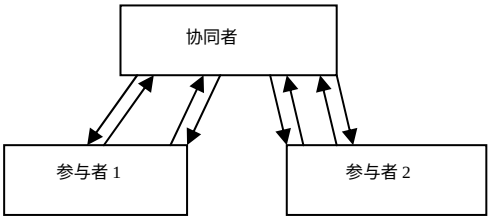

图 6 P3PC 中 3 个节点的位置跟踪消息交换

表 1 P3PC 与 3PC 的额外开销比较

P3PC		3PC（跟踪消息）
传递数据消息	跟踪消息	
6	4(n-1)	2n(n-1)
4n+2		

3 结论

基于代理的 3PC 提交协议可以不关心新的协同者的选举问题，参与者不必保存其它参与者的信息列表，从而大大降低正常处理事务时的额外开销。同时，协同者恢复机制是无缝的转接，可以避免恢复的延时，从而可提供不间断的可靠服务。此外，在事务提交时，绝大部分工作由代理协同者完成，主协同者只负责从代理协同者同步接收数据，有利于维护主协同者的稳定性和可靠性。

参考文献

1 Andrew S T. Distribute Systems Principles and Paradiams[M]. 北京: 清华大学出版社, 2004.

2 Qzsu M T. Principles of Distributed Database Systems[M]. 北京: 清华大学出版社, 2002.

3 陆 风, 郁 梅. 数据安全及双机容错解决方案[J]. 计算机应用研究, 2000, 16(2): 51-52.

4 胡宝霞, 刘丕娥. 容错系统中带检测点的向前运行恢复机制[J]. 电机与控制学报, 1997, 1(3): 150-153.

5 赵高峰, 胡运发. 基于心跳技术的 3 阶段提交协议[J]. 计算机工程与应用, 2004, 40(11): 177-179.

6 廖国琼, 刘云生. 移动实时提交协议的恢复处理与通信分析[J]. 计算机工程与应用, 2004, 40(23): 24-25.

7 Andrew S T. 计算机网络[M]. 4 版. 潘爱民, 译. 北京: 清华大学出版社, 2004.